Amendments to the Specification:

Amend the TITLE of the application as follows:

METHOD, SYSTEM AND COMPUTER PROGRAM PRODUCT TO PARTITION FILTER RULES ~~FOR EFFICIENT ENFORCEMENT~~

Amend page 1, paragraph beginning at line 4 as follows:

This application is related to, and contains common disclosure with, co-pending and commonly assigned patent applications "Network Processor Processing Complex and Methods", serial number 09/384,691, filed August 27, 1999; "Full Match (FM) Search Algorithm Implementation for a Network Processor", serial number 09/543,531 ~~(attorney docket RAL9-1999-0139)~~; and "Software Management Tree Implementation for a Network Processor", serial number 09/545,100 ~~(attorney docket RAL9-1999-0141)~~. Each co-pending patent application is hereby incorporated by reference into this description as fully as if here represented in full.

Amend page 2, paragraph beginning at line 11, through line17 as follows:

In the communications network the database contains a plurality of rules which are applied against IP (Internet packets) received from the internet and for other frames received from other communication facilities or devices. Each rule is associated with a predetermined action. If a rule fits the received packet or frame, the predetermined action, associated with the rule, is applied to the packet or frame. The predetermined action may

include routing decision, filtering decision, etc. Prior art information relating to Rules databases and usage in security, filtering, etc. are set forth in:

Amend page 7, paragraph beginning at line17 as follows:

It can happen that some rules have ranges in two or more components of a key or, having a range in only one component, might not include all possible values in that one range component. Such rules are called herein "range rules." In general, sets of range rules include some rules that intersect. Also, sets of range rules in general are difficult to administer because it can be not obvious which rules apply to various keys. One method of testing keys relative to such sets of intersecting rules is described in "Software Management Tree Implementation for a Network Processor," serial number 09/545,100 ~~(docket RAL9-1999-0141)~~, that is incorporated herein by reference.

Amend page 8, paragraph beginning at line 4 as follows:

It can also happen that some rules have ranges on only one component of a key, for example, there might be a thousand rules in which every component is exact except for th Destination Port number in every rule, which is in every one of the thousand rules a wildcard component. Let us call a set of such rules with given common wildcard component "almost-exact rules." Given that no two rules are identical, in a set of almost-exact rules, it can be proven that no two of any such rules intersect. One method of testing keys relative to such sets of nonintersecting rules is first hashing fixed components of all the rules, preferably a hash like the geometric hash described in "Hash Function for IP, MAC, and Other Structured Addresses", serial number 09/210,222 ~~(docket RA9-98-056)~~, incorporated herein by reference. Then the key can be hashed in the same way and can be processed by the Full Match Tree (FMT) method set forth above and incorporated herein by reference.

Amend page 10, paragraph beginning at line 9 as follows:

Turning to Figure 4 for the moment, a schematic of the database according to the present invention is shown. The database includes N entries R0 through RN-1. Each entry has sub-fields or components SA (Source Address), DA (Destination Address), SP (Source Port), DP (Destination Port), Protocol Type, and Action. More generally, some databases might include another combination of fields or other fields. As stated previously, databases are generated by an operator at the control point processor subsystem 10 (Figure 1) and downloaded into the network processor. According to the present invention the databases are compiled based upon the characteristics of the sub-field. In essence, rules that are almost-exact are stored in one database with a full match search method. All other rules are stored in another database with another type of search method such as SMT. It should be noted that generating databases is a pre-processing non-realtime procedure whereas testing is a realtime procedure.

Amend page 13, paragraph beginning at line 7 as follows:

Still referring to Figure 2, block 32 of the program indicates the entry block where the process is entered. The program then descends into block 34 where F is made equal to all rules R0, ..., RN-1 (Figure 4). Each rule in the database has components also called sub-fields i=1,2,3...n. The program then proceeds into block 36 whereat i is set equal to 1. This means that the first component hereafter called field in each rule will be examined. The program then proceeds into block 38 whereat the rules are grouped into sets and examined. In block 38, Si, with S representing set, is made equal to F. The program then proceeds into block 40. All the rules in S_i are examined. In block 40, if a rule in S_i has a component other than i (the column that is being examined) that is a range (not exact), then that rule is deleted from S_i. Furthermore in block 40, if a rule in S_i has its component i other than a wildcard component, then that rule is deleted from S_i. The program then

proceeds into block 42. In block 42 the algorithm tests in order of label each rule in the set S_i for intersection (multiple rules matching one key) with any second rule in F. If an intersection is found and if the second rule in F has higher priority, then that rule in S_i is deleted from S_i. In an alternative embodiment, if all rules remaining in S_i already have priority number 1 (so dominate all rules with which they intersect), then Block 42 is deleted. The program then proceeds into block 44 whereat the percentage or "fraction" fi of rules remaining in S_i divided by the total number N of rules in F is determined. The program then proceeds into block 46 whereat the component index i is compared to the number of components n. If i is less than n [[T]], then the process increments by setting i equal to i+1. The program then repeats steps in block 38 through block 46.

Amend page 17, paragraph beginning at line 1 as follows:

To generate S1 the values in field 1 of the Rules set must be wild card and other fields must be exact. As is evident by looking at the database, all the rules must be deleted from S1, as shown. The fraction f1 is then generated and since we started off with four rules and none remains, the percentage f1 is 0 percent. It should be noted that the threshold for partitioning the database would typically be set much higher than 25 percent. Therefore, in the first field label choice (field 1), it would not be productive to partition the database. A similar result holds for the S2 set. The process of ignoring a column and testing which rule has all exact values in the sub-field is continued until S3, S4 and S5 are obtained. The percentages are also determined and f3 is equal to 75 percent, f4 0 percent and f5 equal to 0 percent. Suppose the threshold is 75 percent. Because only [[f3]] S3 meets the threshold requirement, the rules associated in S3, namely { R0, R1, R2 }, are placed into one database S_3 and are processed with an FM algorithm. The other rule remaining, R3, is processed with some other method such as an SMT algorithm.